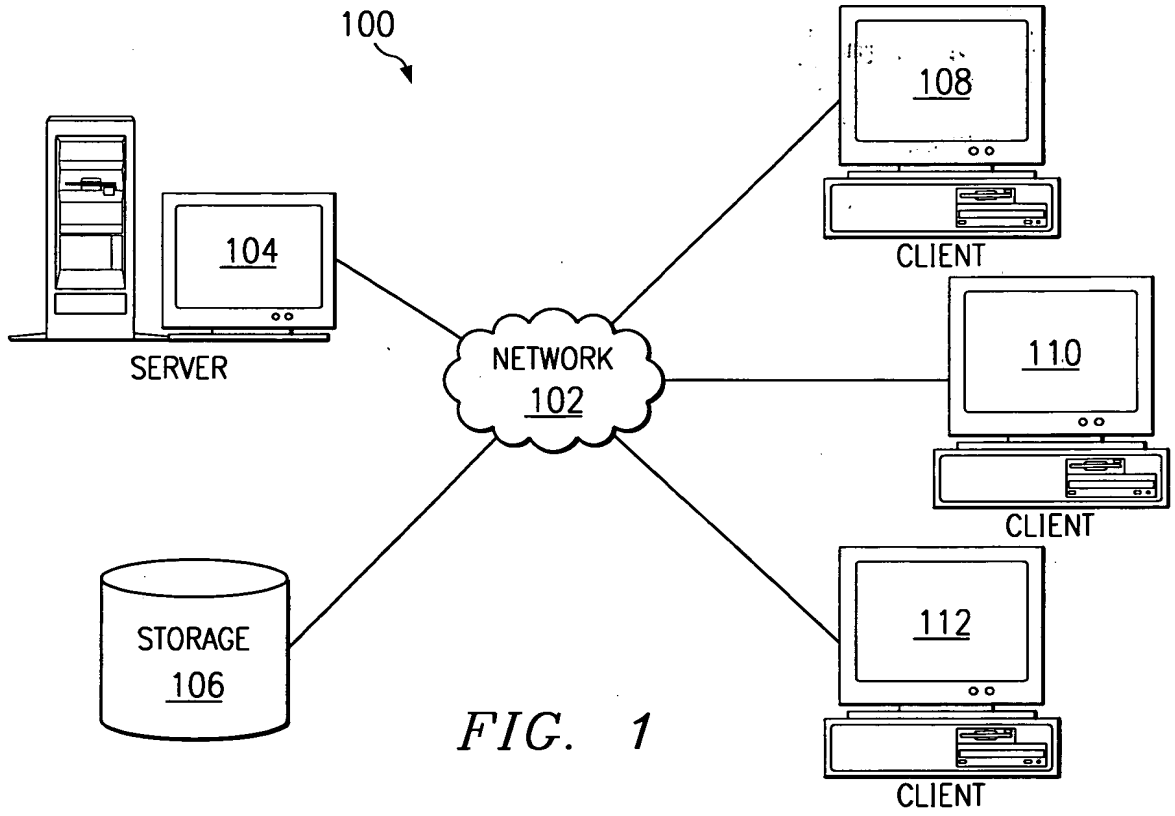
100
108
CLIENT
104
SERVER
NETWORK
102
110
CLIENT
STORAGE
106
112
CLIENT

FIG. 1

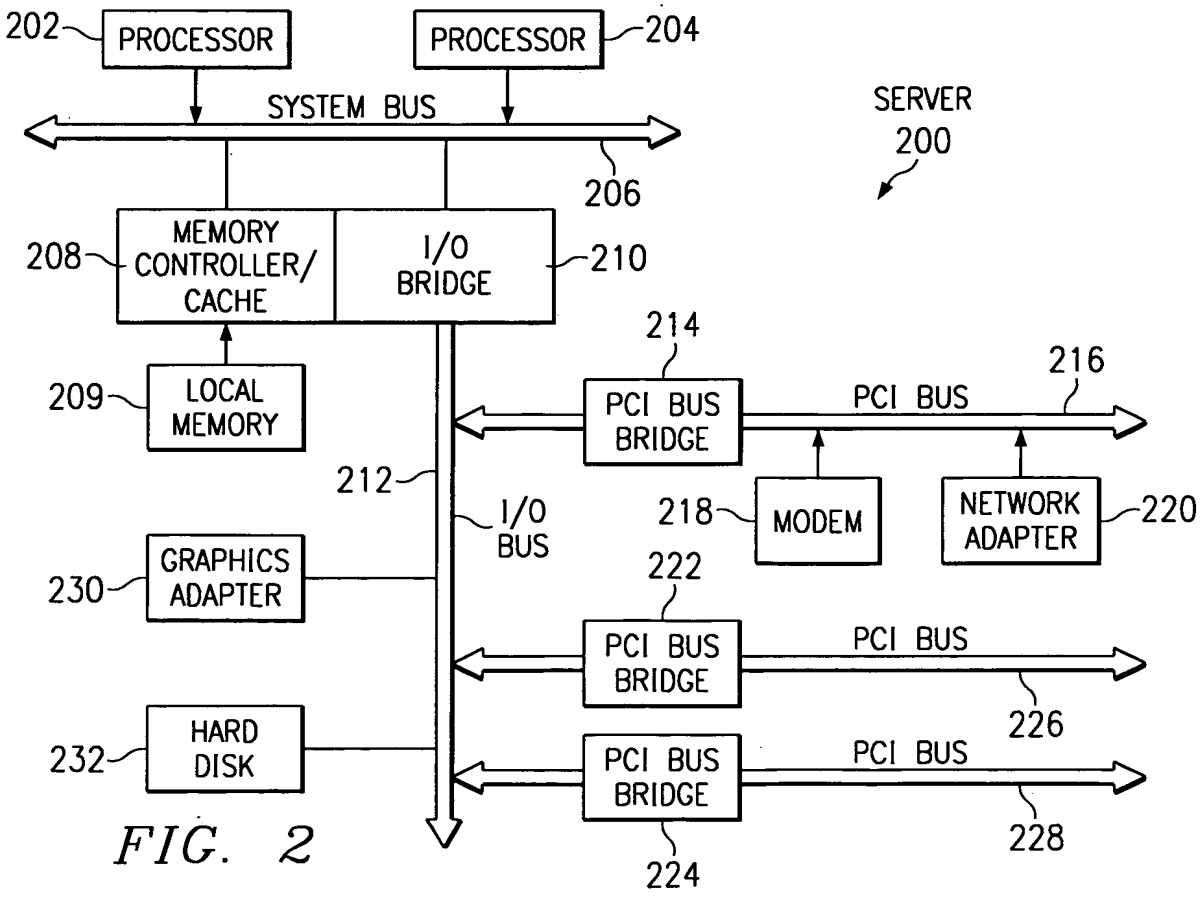
202
PROCESSOR
PROCESSOR
204
SYSTEM BUS
206
SERVER
200
208
MEMORY CONTROLLER/ CACHE
I/O BRIDGE
210
214
216
209
LOCAL MEMORY
PCI BUS BRIDGE
PCI BUS
212
I/O BUS
218
MODEM
NETWORK ADAPTER
220
222
230
GRAPHICS ADAPTER
PCI BUS BRIDGE
PCI BUS
226
232
HARD DISK
PCI BUS BRIDGE
PCI BUS
228
224

FIG. 2

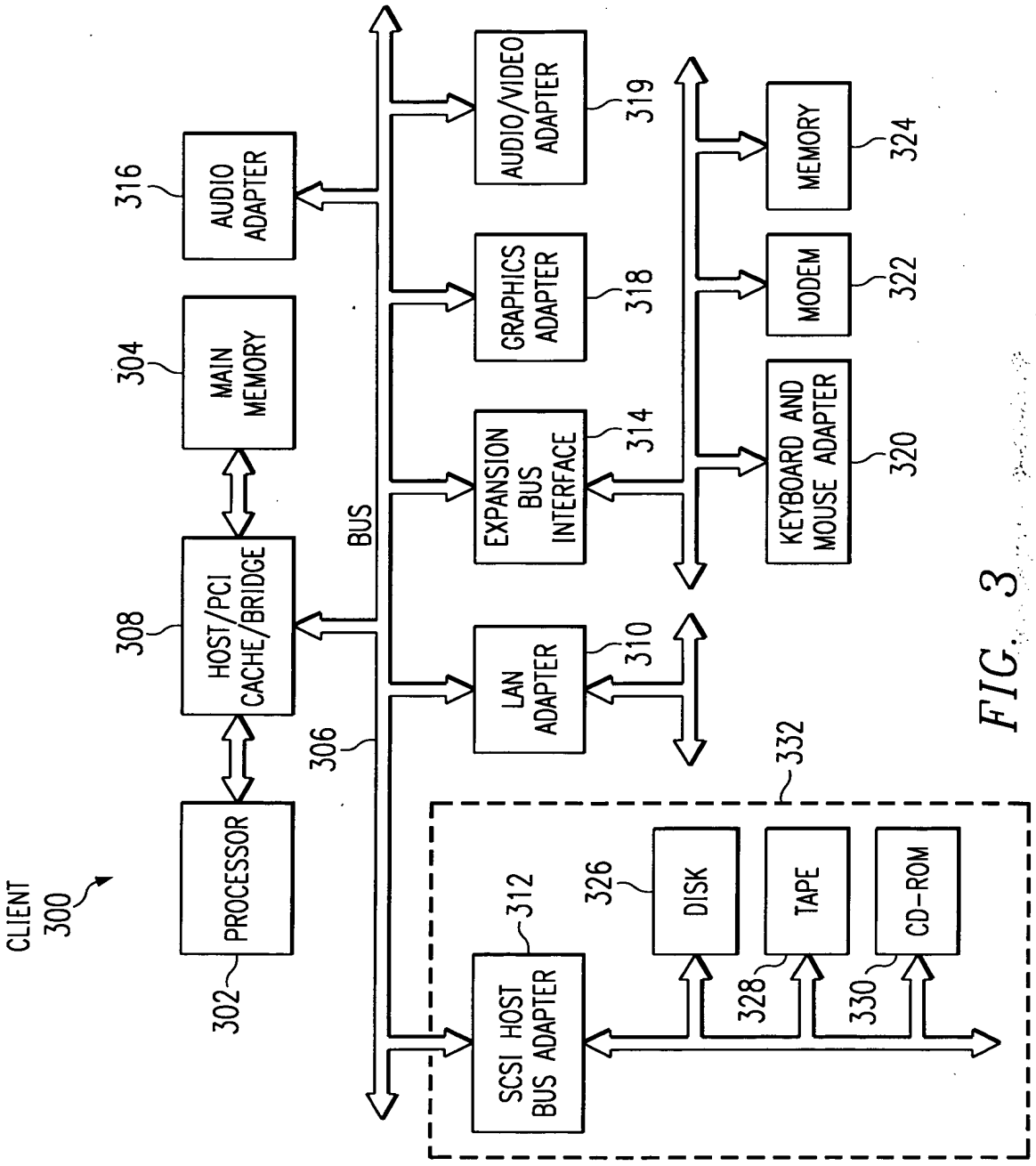
CLIENT
300
PROCESSOR
302
HOST/PCI CACHE/BRIDGE
308
MAIN MEMORY
304
AUDIO ADAPTER
316
BUS
306
LAN ADAPTER
310
EXPANSION BUS INTERFACE
314
GRAPHICS ADAPTER
318
AUDIO/VIDEO ADAPTER
319
KEYBOARD AND MOUSE ADAPTER
320
MODEM
322
MEMORY
324
332
SCSI HOST BUS ADAPTER
312
DISK
326
TAPE
328
CD-ROM
330

FIG. 3

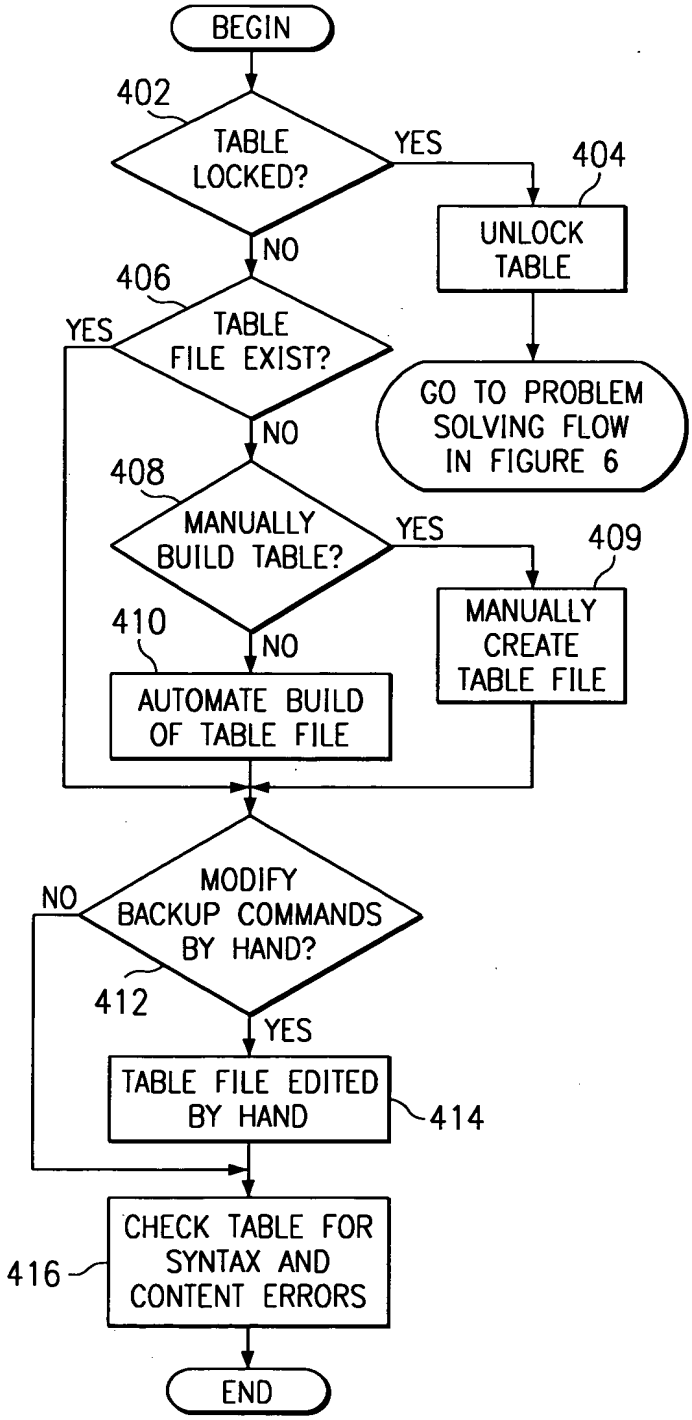
BEGIN
402
TABLE LOCKED?
YES
404
UNLOCK TABLE
NO
406
TABLE FILE EXIST?
YES
GO TO PROBLEM SOLVING FLOW IN FIGURE 6
NO
408
MANUALLY BUILD TABLE?
YES
409
MANUALLY CREATE TABLE FILE
410
NO
AUTOMATE BUILD OF TABLE FILE
NO
MODIFY BACKUP COMMANDS BY HAND?
412
YES
TABLE FILE EDITED BY HAND
414
CHECK TABLE FOR SYNTAX AND CONTENT ERRORS
416
END

FIG. 4

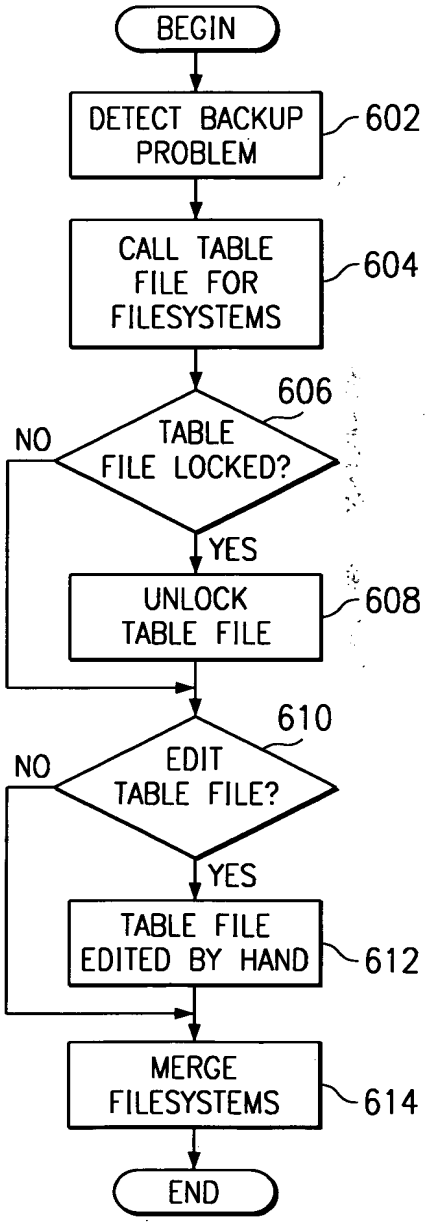
BEGIN
DETECT BACKUP PROBLEM
602
CALL TABLE FILE FOR FILESYSTEMS
604
606
NO
TABLE FILE LOCKED?
YES
UNLOCK TABLE FILE
608
610
NO
EDIT TABLE FILE?
YES
TABLE FILE EDITED BY HAND
612
MERGE FILESYSTEMS
614
END

FIG. 6

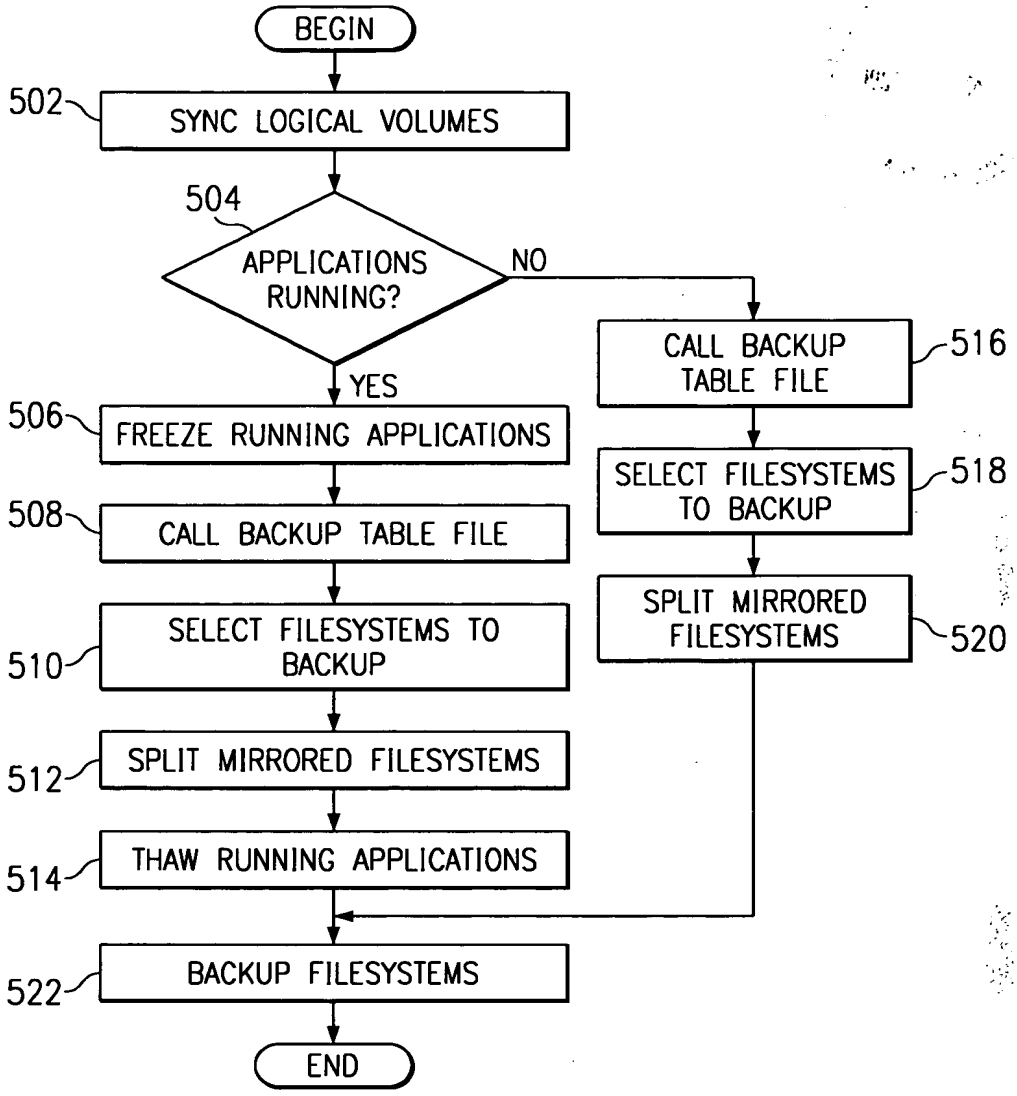

FIG. 5

```
fi

# Table file format
#   Format: bc:pfs:plv:c:afs:alv                    =
#   xb:/home:hd1:2/alt/home:/altlvh                   =


exec 3<&-

-
```

FIG. 9G

```
#!/bin/ksh
###############################################################################
#
# fscpbtab_unlock.ksh
#               Version 0.01
#                       Runs various AIX commands to remove lock on
#                         the FSCPBK table file
#                       Assembled by Carl Gusler
#                        IBM Global Services
#                        IBM Austin
#                        cgusler@us.ibm.com
#
#                               (With help from many friends)
#
#                       Copyright IBM 1996, 1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                        rules, regulations, and retributions of
#                        IBM Global Services
#                        Intellectual Capital Management
#
#
###############################################################################

#------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                rules, and regulations of
#                IBM Global Services
#                Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                described in the IBM Customer Agreement and
#                relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                transfer any rights for replication or distribution
#                of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                course of system maintenance.
#                       2. Copying the program to a similar machine
#                within the same enterprise.
#
#               The customer agrees to restrict access to this
#                program as they would their own proprietary code,
#                and to notify IBM should unauthorized distribution
#                occur.
#
```

FIG. 7A

```
#            This program is distributed on an "as is" basis,
#             no warranty is expressed or implied.
#
#-------------------------------------

#-------------------------------------
#
# Description: Removes lock on /etc/fscpbktab table file.
#         A cleanup utility for problem times with FSCPBK scripts
# Operational Environment: AIX V4
# Input:
# Output:
# Return Value:
# Comments: NOTE!!: This script is an excerpt of the fscpbk_back.ksh
#              script. If that script is edited, this one
#              should probably be edited to match.
#
#-------------------------------------

#-------------------------------------
#
# Version History: None
#
#-------------------------------------

#-------------------------------------
#
# Environmental Variables
#
#-------------------------------------
# Constants
bar='============================================================
'
wire='---------------------------------------'

# Variables
numeric_date=$(date +%m%d%y)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i merge_return_code
typeset -i retain_days=90
typeset -i in_retain_days
typeset -i copies
typeset -i ncrement
typeset -i mount_fs_test
invoked_name=$0
script_name=${invoked_name##*/}
user_id=$(whoami)
desc='ADSM Archive at'$text_date
level=0
```

FIG. 7B

```
# Process Control Variables
l_flag=0
L_flag=0
r_flag=0
d_flag=0

# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
default_backup_device=/dev/rmt0.1
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab
audit_file=/etc/fscpbktab.audit
lock_file=/var/locks/fscpbktab

#-------------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return. It completely exits.
#
#-------------------------------------
show_usage ()
{
  print -u2 "       "
  print -u2 "Usage: fscpbktab_unlock.ksh [-l directory] [-r days] "
  print -u2 "     "
  print -u2 "     -l directory Log output directory."
  print -u2 "                  Default is" $default_log_dir
  print -u2 "       "
  print -u2 "     -r days    Log retention period."
  print -u2 "                  Default is" $retain_days
  print -u2 "     "
  exit 2
}

#-------------------------------------
#
# Korn Shell Settings
#
#-------------------------------------
#set -o errexit    # Turn on error trapping and error exit mode
#set -o noclobber    # Prevent overwriting of existing files
#set -o noexec     # Perform syntax checking without execution
#set -o nolog     # Prevents storing function defs in history file
```

FIG. 7C

```
#set -o xtrace      # Turn on debug mode

#-------------------------------------------------
#
# Main Routine
#
#-------------------------------------------------
#
# Test for any passed parameters.
#if [ $? != 0 ]
#then
#   show_usage
#fi
#
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts l:r# c
do
   case $c in
   l)  # Set up the -l flag
       l_flag=1
       log_dir=$OPTARG;;
   r)  # Set up the -r flag
       r_flag=1
       in_retain_days=$OPTARG;;
   :)  show_usage;;
   \?) show_usage;;
   esac
done
shift $((OPTIND-1))

# Deal with invocation errors
if [[ $user_id != root ]]; then
    show_usage
fi


# Configure Logging
if [[ $l_flag -eq 1 ]]; then
    log_file=$in_log_dir/$default_log_file
    mkdir -p $in_log_dir 2>/dev/null  #Create new log directory
else
    log_file=$default_log_dir/$default_log_file
    mkdir -p $default_log_dir 2>/dev/null # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
 retain_days=$in_retain_days
fi
```

FIG. 7D

```
# Clear old logs
find $log_dir -name "$script_name*" -mtime $retain_days -exec rm {}\;

# Create new log file
exec 3>> $log_file # Open log file for writing

print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log            ="
print -u3 "=                                               ="
print -u3 "=  Created by script:" $script_name
print -u3 "=        on system:" $(hostname)
print -u3 "=        at      :" $(date)
print -u3 "=                                              ="
print -u3 "================================================="

# Perform Work
#  Comments: NOTE!!: This script is an excerpt of the fscpbk_back.ksh
#                    script. If that script is edited, this one
#                    should probably be edited to match.
#

# Test for existing table file
if [[ ! (-r $config_file) ]]; then
    print -u2 "Fatal Table error. Table file" $config_file "not found."
    print -u3 "Fatal Table error. Table file" $config_file "not found."
    exec 3<&-
    exit 99
fi

# Unlock table file

chmod 644 $config_file
rm $lock_file 2>> $log_file


exec 3<&-


exit 0
-
```

FIG. 7E

```
#!/bin/ksh
###############################################################################
#
# fscpbktab_build.ksh
#               Version 0.33
#                       Runs various AIX commands to build
#                         table of filesystems to backup
#                       Assembled by Carl Gusler
#                        IBM Global Services
#                        IBM Austin
#                        cgusler@us.ibm.com
#
#                               (With help from many friends)
#
#                       Copyright IBM 1996, 1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                        rules, regulations, and retributions of
#                        IBM Global Services
#                        Intellectual Capital Management
#
#
###############################################################################

#--------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                rules, and regulations of
#                IBM Global Services
#                Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                described in the IBM Customer Agreement and
#                relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                transfer any rights for replication or distribution
#                of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                course of system maintenance.
#                       2. Copying the program to a similar machine
#                within the same enterprise.
#
#               The customer agrees to restrict access to this
#                program as they would their own proprietary code,
#                and to notify IBM should unauthorized distribution
#                occur.
#
```

FIG. 8A

```
#            This program is distributed on an "as is" basis,
#             no warranty is expressed or implied.
#
#-------------------------------------

#-------------------------------------
#
# Description: Builds table file for other scripts in FSCPBK package.
# Operational Environment: AIX V4 and ADSM V3.1
# Input:
# Output:
# Return Value:
# Comments:
#
#
#-------------------------------------

#-------------------------------------
#
# Version History: None
#
#-------------------------------------

#-------------------------------------
#
# Environmental Variables
#
#-------------------------------------
# Constants
bar='============================================================
'
wire='=-----------------------------------'

# Variables
numeric_date=$(date +%Y%m%d%H%M)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i retain_days=10
typeset -i in_retain_days
typeset -i copies
typeset -i ncrement
typeset -i return_code
invoked_name=$0
script_name=${invoked_name##*\}
user_id=$(whoami)


# Process Control Variables
l_flag=0
L_flag=0
r_flag=0
```

FIG. 8B

```
# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab
lock_file=/var/locks/fscpbktab

#-------------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return. It completely exits.
#
#-------------------------------------
show_usage ()
{
   print -u2 "       "
   print -u2 "Usage: fscpbktab_build.ksh [-l directory] [-r days] "
   print -u2 "       "
   print -u2 "       -l directory   Log output directory."
   print -u2 "                      Default is" $default_log_dir
   print -u2 "       "
   print -u2 "       -r days    Log retention period."
   print -u2 "                  Default is" $retain_days
   print -u2 "       "
   exit 2
}

#-------------------------------------
#
# Korn Shell Settings
#
#-------------------------------------
#set -o errexit    # Turn on error trapping and error exit mode
#set -o noclobber    # Prevent overwriting of existing files
#set -o noexec     # Perform syntax checking without execution
#set -o nolog      # Prevents storing function defs in history file
#set -o xtrace     # Turn on debug mode

#-------------------------------------
#
# Main Routine
#
#-------------------------------------
```

FIG. 8C

```
#
# Test for any passed parameters.
#if [ $? != 0 ]
#then
# show_usage
#fi
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts a:l:p:r# c
do
   case $c in
   l)    # Set up the -l flag
         l_flag=1
         log_dir=$OPTARG;;
   r)    # Set up the -r flag
         r_flag=1
         in_retain_days=$OPTARG;;
   :)    show_usage;;
   \?)   show_usage;;
   esac
done
shift $((OPTIND-1))

# Deal with invocation errors
if [[ $user_id != root ]]; then
    show_usage
fi


# Configure Logging
if [[ $l_flag -eq 1 ]]; then
    log_file=$in_log_dir/$default_log_file
    mkdir -p $in_log_dir 2>/dev/null  #Create new log directory
else
    log_file=$default_log_dir/$default_log_file
    mkdir -p $default_log_dir 2>/dev/null # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
    retain_days=$in_retain_days
fi

# Clear old logs
find $log_dir -name "$script_name*" -mtime $retain_days -exec rm {}\;

# Create new log file
exec 3>> $log_file # Open log file for writing
```

FIG. 8D

```
print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log             ="
print -u3 "=                                               ="
print -u3 "= Created by script:" $script_name
print -u3 "=       on system:" $(hostname)
print -u3 "=       at    :" $(date)
print -u3 "=                                               ="
print -u3 "================================================="

# Perform Work

# Test for locked table file and exit
if [[ -f $lock_file ]]; then
    print -u2 "Table file is currently in use and locked."
    print -u3 "Table file is currently in use and locked."
    exec 3<&-
    exit 96
fi

# Test for existing table file and save
if [[ -r $config_file ]]; then
    mv $config_file $config_file.old.$text_date
fi


# Create new tab file
exec 4> $config_file # Open table file for writing
#print -u4 "#:"$(date +%Y%m%d%H%M"):"============================="
print -u4 "#================================================="
print -u4 "#                                               ="
print -u4 "# Filesystem Backup Selection Table file        ="
print -u4 "#                                               ="
print -u4 "#  Format: bc:pfs:plv:c:afs:alv                 ="
print -u4 "#                                               ="
print -u4 "#       or                                      ="
print -u4 "#                                               ="
print -u4 "#       bc (Backup Control)                     ="
print -u4 "#         xb -> AIX Backup (Level 0 AIX FS Backup)  ="
print -u4 "#         no -> NO Backup (Skip filesystem)     ="
print -u4 "#         as -> ADSM Selective Backup           ="
print -u4 "#         ai -> ADSM Incremental Backup         ="
print -u4 "#         aa -> ADSM Archive                    ="
print -u4 "#                                               ="
print -u4 "#                                               ="
print -u4 "#       pfs (Primary Filesystem)                ="
print -u4 "#         The full path of standard filesystem  ="
print -u4 "#                                               ="
print -u4 "#       plv (Primary Logical Volume)            ="
```

FIG. 8E

```
                print -u4 "#         The AIX LV name of the logical volume    ="
print -u4 "#           containing the primary filesystem          ="
print -u4 "#                                                      ="
print -u4 "#         c (Copies)                                   ="
print -u4 "#           The number of AIX LVM copies of the        ="
print -u4 "#           logical volume containing primary          ="
print -u4 "#           filesystem.                                ="
print -u4 "#           Must be numeric 1,2, or 3.                 ="
print -u4 "#                                                      ="
print -u4 "#         afs (Alternate Filesystem)                   ="
print -u4 "#           The full path of mirror copy filesystem    ="
print -u4 "#           Must be unique!!!!!                        ="
print -u4 "#                                                      ="
print -u4 "#         alv (Alternate Logical Volume)               ="
print -u4 "#           The AIX LV name of the logical volume      ="
print -u4 "#           containing the alternate filesystem        ="
print -u4 "#           Must be unique!!!!!                        ="
print -u4 "#                                                      ="
print -u4 "#   Example for a mirrored home filesystem to be       ="
print -u4 "#      backed up using AIX backup command:             ="
print -u4 "#                                                      ="
print -u4 "#   xb:/home:hd1:2:/alt/home:altlvh                     ="
print -u4 "#                                                      ="
print -u4 "#=========================================================="

print -u3 "\nStarting Build of Filesystem Backup Table File."
print -u3 "\nTable lines are:"
ncrement=0
return_code=0
for fs_line in $(lsfs -ac | grep -v ^#)
do
  if [[ $(print $fs_line | cut -f 3 -d : ) = jfs ]]; then
  fs_prime=$(print $fs_line | cut -f 1 -d :)
  lv_prime=$(print $fs_line | cut -f 2 -d : | cut -c 6-)

# What if LV in /etc/filesystems does not actually exist?
#  LSLV below crooks

  copies=$(lslv $lv_prime | grep COPIES | awk '{ print $2 }')
  if [[ $copies -eq 1 ]]; then
      tab_line=xb:$fs_prime:$lv_prime:$copies
  elif [[ $copies -eq 2 ]]; then
      tab_line=xb:$fs_prime:$lv_prime:$copies:/alt/fs$ncrement:altlv$ncrement
      ((ncrement=$ncrement+1))
  elif [[ $copies -eq 3 ]]; then
      tab_line=xb:$fs_prime:$lv_prime:$copies:/alt/fs$ncrement:altlv$ncrement
      ((ncrement=$ncrement+1))
  else
```

FIG. 8F

```
      tab_line=xb:$fs_prime:$lv_prime:1
      print -u2 "Script execution error: AIX lslv output confusion."
      print -u3 "Script execution error: AIX lslv output confusion."
      ((return_code=$return_code+1))
   fi
   print -u3 $tab_line
   print -u4 $tab_line
   fi
done


exec 3<&-
exec 4<&-

# Test for filesystem parsing problems
if [[ $return_code -ne 0 ]]; then
    exit 10
fi

exit 0
-
```

FIG. 8G

FIG. 12J

```
            else
              print -u3 "Filesystem" $target_fs "not mountable. Not backed up!"
              return_code=1
            fi
         fi
      done


      exec 3<&-

      # Test for unsuccessful filesystem merges
      if [[ $merge_return_code -ne 0 ]]; then
          exit 20
      fi

      rm $lock_file 2>/dev/null
      chmod 644 $config_file

      # Test for unsuccessful filesystem backups
      if [[ $return_code -ne 0 ]]; then
          exit 10
      fi

      exit 0
      -
```

```
#!/bin/ksh
###############################################################################
#
# fscpbktab_check.ksh
#               Version 0.33
#                       Runs various AIX commands to check filesystem
#                         table file
#                       Assembled by Carl Gusler
#                        IBM Global Services
#                        IBM Austin
#                        cgusler@us.ibm.com
#
#                               (With help from many friends)
#
#                       Copyright IBM 1996, 1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                        rules, regulations, and retributions of
#                        IBM Global Services
#                        Intellectual Capital Management
#
#
###############################################################################

#-------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                rules, and regulations of
#                IBM Global Services
#                Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                described in the IBM Customer Agreement and
#                relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                transfer any rights for replication or distribution
#                of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                course of system maintenance.
#                       2. Copying the program to a similar machine
#                within the same enterprise.
#
#               The customer agrees to restrict access to this
#                program as they would their own proprietary code,
#                and to notify IBM should unauthorized distribution
#                occur.
#
```

FIG. 9A

```
#           This program is distributed on an "as is" basis,
#            no warranty is expressed or implied.
#
#-------------------------------------

#-------------------------------------
#
# Description: Performs syntax check on FSCPBK table file.
#         Part of FSCPBK package of scripts.
# Operational Environment: AIX V4 and ADSM V3.1
# Input:
# Output:
# Return Value:
# Comments:
#
#
#-------------------------------------

#-------------------------------------
#
# Version History: None
#
#-------------------------------------

#-------------------------------------
#
# Environmental Variables
#
#-------------------------------------
# Constants
bar='=================================================
'
wire='-----------------------------------------'

# Variables
numeric_date=$(date +%m%d%y)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i retain_days=90
typeset -i in_retain_days
typeset -i copies
typeset -i lv_copies
typeset -i lv_disks
typeset -i ncrement
typeset -i return_code
invoked_name=$0
script_name=${invoked_name##*/}
user_id=$(whoami)
```

FIG. 9B

```
# Process Control Variables
l_flag=0
L_flag=0
r_flag=0

# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab
audit_file=/etc/fscpbktab.audit
lock_file=/var/locks/fscpbktab

#-------------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return. It completely exits.
#
#-------------------------------------
show_usage ()
{
   print -u2 "       "
   print -u2 "Usage: fscpbktab_check.ksh [-l directory] [-r days]"
   print -u2 "       "
   print -u2 "       -l directory Log output directory."
   print -u2 "          Default is" $default_log_dir
   print -u2 "       "
   print -u2 "       -r days    Log retention period."
   print -u2 "                  Default is" $retain_days
   print -u2 "       "
   exit 2
}

#-------------------------------------
#
# Korn Shell Settings
#
#-------------------------------------
#set -o errexit    # Turn on error trapping and error exit mode
#set -o noclobber  # Prevent overwriting of existing files
#set -o noexec     # Perform syntax checking without execution
#set -o nolog   # Prevents storing function defs in history file
#set -o xtrace   # Turn on debug mode
```

FIG. 9C

```
#-----------------------------------------
#
# Main Routine
#
#-----------------------------------------
#
# Test for any passed parameters.
#if [ $? != 0 ]
#then
#   show_usage
#fi
#
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts a:l:p:r# c
do
   case $c in
   l)  # Set up the -l flag
       l_flag=1
       log_dir=$OPTARG;;
   r)  # Set up the -r flag
       r_flag=1
       in_retain_days=$OPTARG;;
   :)  show_usage;;
   \?) show_usage;;
   esac
done
shift $((OPTIND-1))

# Deal with invocation errors

# Configure Logging
if [[ $l_flag -eq 1 ]]; then
    log_file=$in_log_dir/$default_log_file
    mkdir -p $in_log_dir 2>/dev/null  #Create new log directory
else
    log_file=$default_log_dir/$default_logfile
    mkdir -p $default_log_dir 2>/dev/null # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
    retain_days=$in_retain_days
fi

# Clear old logs
find $log_dir -name "$script_name*" -mtime $retain_days -exec rm {}\;

# Create new log file
exec 3>> $log_file # Open log file for writing
```

FIG. 9D

```
print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log            ="
print -u3 "=                                               ="
print -u3 "=  Created by script:" $script_name
print -u3 "=        on system:" $(hostname)
print -u3 "=        at    :" $(date)
print -u3 "=                                               ="
print -u3 "================================================="

# Perform Work

# Test for existing table file
if [[ ! (-r $config_file) ]]; then
        print -u2 "Table error: Table file" $config_file "does not exist."
        print -u3 "Table error: Table file" $config_file "does not exist."
    exit 99
fi

# Test for locked table file
if [[ -f $lock_file ]]; then
    print -u2 "Warning: Table file is currently in use and locked."
    print -u3 "Warning: Table file is currently in use and locked."
fi

# Perform Syntax Checking on Table File
return_code=0
ncrement=1
for fs_line in $(cat $config_file | grep -v ^#)
do
    action=$(print $fs_line | cut -f 1 -d :)
    case $action in
      xb) : ;;
      no) : ;;
      as) : ;;
      ai) : ;;
      aa) : ;;
       *) print -u2 "Table error: Action" $action "not valid."
          print -u3 "Table error: Action" $action "not valid."
          ((return_code=$return_code+1));;
    esac
    fs_prime=$(print $fs_line | cut -f 2 -d :)
    lv_prime=$(print $fs_line | cut -f 3 -d :)
    if [[ $(lsfs -c $fs_prime | grep $lv_prime | wc -l) -ne 1 ]]; then
        print -u2 "Table error: Filesystem" $fs_prime "does not reside in LV $lv_prime
        print -u3 "Table error: Filesystem" $fs_prime "does not reside in LV $lv_prime
          ((return_code=$retum_code+1))
    fi
    copies=$(print $fs_line | cut -f 4 -d :)
```

FIG. 9E

FIG. 9F

```
if [[ ($copies -ge 1) && ($copies -le 3) ]]; then
   if [[ ($copies -gt 1) && ($copies -le 3) ]]; then
      fs_alt=$(print $fs_line | cut -f 5 -d :)
      lv_alt=$(print $fs_line | cut -f 6 -d :)
      if [[ $(lsfs -c $fs_alt 2>/dev/null | wc -l) -ne 0 ]]; then
          print -u2 "Table error: Filesystem" $fs_alt "already exists."
          print -u3 "Table error: Filesystem" $fs_alt "already exists."
          ((return_code=$return_code+1))
      fi
      if [[ $(lslv $lv_alt 2>/dev/null | wc -l) -ne 0 ]]; then
          print -u2 "Table error: LV" $lv_alt "already exists."
          print -u3 "Table error: LV" $lv_alt "already exists."
          ((return_code=$return_code+1))
      fi
      strictness_flag=$(lslv $lv_prime | grep "EACH LP COPY ON" | grep yes | wc -l)
      if [[ $strictness_flag -eq 0 ]]; then
          print -u2 "LVM Warning: Mirror strictness not set for LV" $lv_prime
          print -u3 "LVM Warning: Mirror strictness not set for LV" $lv_prime
      fi
      lv_copies=$(lslv $lv_prime | grep COPIES | awk'{ print $2 }')
      if [[ $lv_copies -ne $copies ]]; then
          print -u2 "LVM Warning: LV mirroring does not match table for LV" $lv_prime
          print -u3 "LVM Warning: LV mirroring does not match table for LV" $lv_prime
      fi
      lv_disks=$(lslv -l $lv_prime | grep hdisk | wc -l)
      if [[ $lv_disks -ne $lv_copies ]]; then
          print -u2 "LVM Warning: Broad LV mirroring distribution for LV" $lv_prime
          print -u3 "LVM Warning: Broad LV mirroring distribution for LV" $lv_prime
      fi
    fi
   else
      print -u2 "Table error: Invalid number of LV copies for LV" $lv_prime
      print -u3 "Table error: Invalid number of LV copies for LV" $lv_prime
      ((return_code=$return_code+1))
   fi
done

if [[ ($return_code -ne 0) ]];then
 return 98
else
 print -u2 "Table file parses okay."
 exec 4> $audit_file # Open audit file for writing
 current_Y=$(date +%Y)
 current_m =$(date +%m)
 current_d=$(date +%d)
 current_H =$(date +%H)
 current_M=$(date +%M)
#    print -u4 $current_Y $current_m $current_d $current_H $current_M
    print -u4 $current_Y$current_m$current_d$current_H$current_M
    exec 4<&-
```

```
#!/bin/ksh
###############################################################################
#
# fscpb_sync.ksh
#               Version 0.02
#                       Runs various AIX commands to synchronize all
#                         stale logical volumes
#                       Assembled by Carl Gusler
#                         IBM Global Services
#                         IBM Austin
#                         cgusler@us.ibm.com
#
#                                 (With help from many friends)
#
#                       Copyright IBM 1996, 1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                         rules, regulations, and retributions of
#                         IBM Global Services
#                         Intellectual Capital Management
#
#
###############################################################################

#--------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                 rules, and regulations of
#                 IBM Global Services
#                 Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                 described in the IBM Customer Agreement and
#                 relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                 transfer any rights for replication or distribution
#                 of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                 course of system maintenance.
#                       2. Copying the program to a similar machine
#                 within the same enterprise.
#
#               The customer agrees to restrict access to this
#                 program as they would their own proprietary code,
#                 and to notify IBM should unauthorized distribution
#                 occur.
#
```

FIG. 10A

```
#              This program is distributed on an "as is" basis,
#                no warranty is expressed or implied.
#
#-------------------------------------

#-------------------------------------
#
# Description: Synchronizes all logical volumes with stale partitions
#         Part of FSCPBK package.
# Operational Environment: AIX V4
# Input:
# Output:
# Return Value:
# Comments:
#
#
#-------------------------------------

#-------------------------------------
#
# Version History: None
#
#-------------------------------------

#-------------------------------------
#
# Environmental Variables
#
#-------------------------------------
# Constants
bar='============================================================
'
wire='----------------------------------------'

# Variables
numeric_date=$(date +%m%d%y)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i retain_days=90
typeset -i in_retain_days
typeset -i copies
typeset -i ncrement
typeset -i return_code
invoked_name=$0
script_name=${invoked_name##*/}
user_id=$(whoami)
```

FIG. 10B

```
# Process Control Variables
l_flag=0
L_flag=0
r_flag=0

# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab

#-----------------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return.  It completely exits.
#
#-----------------------------------------
show_usage ()
{
   print -u2 "       "
   print -u2 "Usage: fscpbk_sync.ksh [-l directory] [-r days] "
   print -u2 "       "
   print -u2 "       -l directory  Log output directory."
   print -u2 "                     Default is $default_log_dir
   print -u2 "       "
   print -u2 "       -r days       Log retention period."
   print -u2 "                     Default is" $retain_days
   print -u2 "       "
   exit 2
}

#-----------------------------------------
#
# Korn Shell Shell Settings
#
#-----------------------------------------
#set -o errexit    #Turn on error trapping and error exit mode
#set -o noclobber   # Prevent overwriting of existing files
#set -o noexec     # Perform syntax checking without execution
#set -o nolog     # Prevents storing function defs in history file
#set -o xtrace   # Turn on debug mode

#-----------------------------------------
#
```

FIG. 10C

```
# Main Routine
#
#----------------------------------------
#
# Test for any passed parameters.
#if [ $? != 0 ]
#then
# show_usage
#fi
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts l:r# c
do
   case $c in
   l)  # Set up the -l flag
       l_flag=1
       log_dir=$OPTARG;;
   r)  # Set up the -r flag
       r_flag=1
       in_retain_days=$OPTARG;;
   :)  show_usage;;
   \?) show_usage;;
   esac
done
shift $((OPTIND-1))

# Deal with invocation errors
if [[ $user_id != root ]]; then
  show_usage
fi


# Configure Logging
if [[ $l_flag -eq 1 ]]; then
    log_file=$in_log_dir/$default_log_file
    mkdir -p $in_log_dir 2>/dev/null  #Create new log directory
else
    log_file=$default_log_dir/$default_log_file
    mkdir -p $default_log_dir 2>/dev/null # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
  retain_days=$in_retain_days
fi

# Clear old logs
find $log_dir -name "$script_name*" -mtime $retain_days -exec rm {}\;
```

FIG. 10D

```
# Create new log file
exec 3>> $log_file # Open log file for writing

print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log            ="
print -u3 "=                                               ="
print -u3 "=   Created by script:" $script_name
print -u3 "=         on system:" $(hostname)
print -u3 "=         at      :" $(date)
print -u3 "=                                               ="
print -u3 "================================================="

# Perform Work

# Test for any stale logical volumes within active volume groups

print -u1 "Starting syncvg operation. This make take several minutes."
return_code=0
for logical_volume in $(lsvg -o | lsvg -il | grep stale | awk'{ print $1 }')
do
   print -u3 " Starting syncvg operation on LV" $logical_volume
   print -u1 "Starting syncvg operation on LV" $logical_volume
   syncvg -l $logical_volume
   ((return_code=$return_code+$?))
   print -u3 " Completed syncvg operation on LV  $logical_volume
   print -u3 " Cumulated return code is" $return"code
done

exec 3<&-
if [[ ($return_code -ne 0) ]];then
    return 50
fi

exit 0

-
```

FIG. 10E

```
#!/bin/ksh
###############################################################################
#
# fscpb_select.ksh
#               Version 0.34
#                       Runs various AIX commands to select and split
#                         filesystems for backup
#                       Assembled by Carl Gusler
#                         IBM Global Services
#                         IBM Austin
#                         cgusler@us.ibm.com
#
#                               (With help from many friends)
#
#                       Copyright IBM 1996, 1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                         rules, regulations, and retributions of
#                         IBM Global Services
#                         Intellectual Capital Management
#
#
###############################################################################

#--------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                 rules, and regulations of
#                 IBM Global Services
#                 Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                 described in the IBM Customer Agreement and
#                 relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                 transfer any rights for replication or distribution
#                 of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                 course of system maintenance.
#                       2. Copying the program to a similar machine
#                 within the same enterprise.
#
#               The customer agrees to restrict access to this
#                 program as they would their own proprietary code,
#                 and to notify IBM should unauthorized distribution
#                 occur.
#
```

FIG. 11A

```
#           This program is distributed on an "as is" basis,
#           no warranty is expressed or implied.
#
#-------------------------------------

#-------------------------------------
#
# Description: Selects and splits filesystems for backup.
#      Part of FSCPBK package of scripts.
# Operational Environment: AIX V4
# Input:
# Output:
# Return Value:
# Comments:
#
#
#-------------------------------------

#-------------------------------------
#
# Version History: None
#
#-------------------------------------

#-------------------------------------
#
# Environmental Variables
#
#-------------------------------------
# Constants
bar='=================================================
'
wire='-----------------------------------------'

# Variables
numeric_date=$(date +%m%d%y)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i retain_days=90
typeset -i in_retain_days
typeset -i copies
typeset -i new_copies
typeset -i ncrement
typeset -i ntest
typeset -i return_code
#typeset -i edit_year
#typeset -i edit_month
#typeset -i edit_day
#typeset -i edit_hour
```

FIG. 11B

```
#typeset -i edit_minute
typeset -i edit_stamp
typeset -i audit_year
typeset -i audit_month
typeset -i audit_day
typeset -i audit_hour
typeset -i audit_minute
typeset -i audit_stamp
invoked_name=$0
script_name=${invoked_name##*/}
user_id=$(whoami)


# Process Control Variables
l_flag=0
L_flag=0
r_flag=0
o_flag=0

# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab
audit_file=/etc/fscpbktab.audit
lock_file=/var/locks/fscpbktab

#-------------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return. It completely exits.
#
#-------------------------------------
show_usage ()
{
   print -u2 "       "
   print -u2 "Usage: fscpbk_select.ksh -o [-l directory] [-r days] "
   print -u2 "       "
   print -u2 "       -o       Override active volume protection."
   print -u2 "                WARNING!!: Data integrity risk."
   print -u2 "                     IBM not responsible for"
   print -u2 "                     loss of data or integrity"
   print -u2 "                     if override used to split"
```

FIG. 11C

```
    print -u2 "              a mirrored filesystem"
    print -u2 "              that is mounted!"
    print -u2 "      "
    print -u2 "      -l directory Log output directory."
    print -u2 "                   Default is" $default_log_dir
    print -u2 "      "
    print -u2 "      -r days     Log retention period."
    print -u2 "                   Default is" $retain_days
    print -u2 "      "
    exit 2
}

#--------------------------------------
#
# Korn Shell Settings
#
#--------------------------------------
#set -o errexit    # Turn on error trapping and error exit mode
#set -o noclobber  # Prevent overwriting of existing files
#set -o noexec     # Perform syntax checking without execution
#set -o nolog      # Prevents storing function defs in history file
#set -o xtrace     # Turn on debug mode

#--------------------------------------
#
# Main Routine
#
#--------------------------------------
#
# Test for any passed paramaters.
#if [ $? != 0 ]
#then
#   show_usage
#fi
#
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts ol:r# c
do
   case $c in
   o)    # Set up the -o flag
         o_flag=1;;
   l)    # Set up the -l flag
         l_flag=1
         log_dir=$OPTARG;;
   r)    # Set up the -r flag
         r_flag=1
         in_retain_days=$OPTARG;;
   :)    show_usage;;
   \?)   show_usage;;
```

FIG. 11D

```
   esac
done
shift $((OPTIND-1))

# Deal with invocation errors
if [[ $user_id != root ]]; then
    show_usage
fi

if [[ $o_flag -ne 1 ]]; then
    show_usage
fi


# Configure Logging
if [[ $l_flag -eq 1 ]]; then
    log_file=$in_log_dir/$default_log_file
    mkdir -p $in_log_dir 2>/dev/null  #Create new log directory
else
    log_file=$default_log_dir/$default_log_file
    mkdir -p $default_log_dir 2>/dev/null  # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
    retain_days=$in_retain_days
fi

# Clear old logs
find $log_dir -name "$script_name*" -mtime $retain_days -exec rm{}\;

# Create new log file
exec 3>> $log_file # Open log file for writing

print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log           ="
print -u3 "=                                               ="
print -u3 "=  Created by script:" $script_name
print -u3 "=       on system:" $(hostname)
print -u3 "=       at      :"$(date)
print -u3 "=                                               ="
print -u3 "================================================="

# Perform Work

# Test for existing table file
if [[ ! (-r $config_file) ]]; then
    print -u2 "Fatal Table error. Table file" $config_file "not found."
```

FIG. 11E

```
    print -u3 "Fatal Table error. Table file" $config_file "not found."
    exec 3<&-
    exit 99
fi

#  Test for existing table audit file
if [[ ! (-r $audit_file) ]]; then
    print -u2 "Fatal Table error. Table file check program must be run."
    print -u3 "Fatal Table error. Table audit file" $audit_file "not found."
    exec 3<&-
    exit 97
fi

#  Test for table file audit indicating syntax check since last edit


current_Y=$(date +%Y)

audit_stamp=$( head -1 $audit_file | awk '{ print $1 }')

#  Check for colon and thus time instead of year on file datestamp
ntest=$(ls -l $config_file | awk'{ print $8 }' | grep : | wc -l)
if [[ $ntest -eq 1 ]]; then
    edit_year=$current_Y
else

   edit_year=$(ls -l $config_file | awk'{ print $8 }')
fi


edit_month_text=$(ls -l $config_file | awk '{ print $6 }')
edit_day=$(ls -l $config_file | awk '{ print $7 }')
edit_hour=$(ls -l $config_file | awk '{ print $8 }' | cut -f 1 -d :)
edit_minute=$(ls -l $config_file | awk '{ print $8 }' | cut -f 2 -d :)

# Determine month number from month name
case $edit_month_text in
Jan)  edit_month=01;;
Feb)  edit_month=02;;
Mar)  edit_month=03;;
Apr)  edit_month=04;;
May)  edit_month=05;;
Jun)  edit_month=06;;
Jul)   edit_month=07;;
Aug)  edit_month=08;;
Sep)  edit_month=09;;
Oct)  edit_month=10;;
Nov)  edit_month=11;;
Dec)  edit_month=12;;
```

FIG. 11F

```
*)    print -u2 "Fatal Table error. Table file date read error."
      print -u3 "Fatal Table error. Table file date read error."
      exec 3<&-
      exit 98;;
esac


edit_stamp=$edit_year$edit_month$edit_day$edit_hour$edit_minute

# Test for table file audited since last editing
if [[ $audit_stamp -le $edit_stamp ]]; then
    print -u2 "Fatal Table error. Table file edited since last checked."
    print -u3 "Fatal Table error. Table file edited since last checked."
    exec 3<&-
    exit 97
fi

# Test for locked table file and exit
if [[ -f $lock_file ]]; then
    print -u2 "Table file is currently in use and locked."
    print -u3 "Table file is currently in use and locked."
    exec 3<&-
    exit 96
fi
```

FIG. 11G

```
# Table file format
# Format: bc:pfs:plv:c:afs:alv                    =
# xb:/home:hd1:2:/alt/home:/altlvh                  =

# Create lock on table file to indicate that table is in use.
touch $lock_file
chmod 000 $config_file


# Increment through table file and split mirrored filesystems
return_code=0
ncrement=0
for fs_line in $(cat $config_file | grep -v ^#)
do
  action=$(print $fs_line | cut -f 1 -d :)
  copies=$(print $fs_line | cut -f 4 -d :)
  if [[ ($copies -gt 1) && ($action != no) ]]; then
      fs_prime=$(print $fs_line | cut -f 2 -d :)
      lv_prime=$(print $fs_line | cut -f 3 -d :)
      fs_alt=$(print $fs_line | cut -f 5 -d :)
      lv_alt=$(print $fs_line cut -f 6 -d :)
      tag_file=$fs_prime/.fscpbk_$lv_prime
      exec 4> $tag_file     # Open tag file for overwriting
```

```
        print -u4 "#================================================="
        print -u4 "#=                                              ="
        print -u4 "#= Tag file used by IBM FSCPBK Utility.        ="
        print -u4 "#= DO NOT DELETE THIS FILE!!!!!!!!!!!!!!!!!!!!  ="
        print -u4 "#=                                              ="
        print -u4 "#= Files in this directory and subdirectories below    ="
        print -u4 "#=  were originally contained within filesystem:    ="
        print -u4 "#=  " $fs_prime
        print -u4 "#=                                              ="
        print -u4 "#================================================="
        exec 4<&-
        ((new_copies=$copies-1))
        sync;sync
        split_fs_copy.ksh -f $fs_prime -n $fs_alt -y $lv_alt -c $new_copies -o
        ((return_code=$return_code+$?))
        print -u3 $action $fs_prime $lv_prime $copies $fs_alt $lv_alt
    fi
done


exec 3<&-

if [[ ($return_code -ne 0) ]];then
    exit 10
else
  exit 0
fi

-
```

FIG. 11H

```
#!/bin/ksh
###############################################################################
#
# fscpb_back.ksh
#               Version 0.34
#                       Runs various AIX commands to backup and merge
#                         filesystems
#                       Assembled by Carl Gusler
#                        IBM Global Services
#                        IBM Austin
#                        cgusler@us.ibm.com
#
#                               (With help from many friends)
#
#                       Copyright IBM 1996,1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                        rules, regulations, and retributions of
#                        IBM Global Services
#                        Intellectual Capital Management
#
#
###############################################################################

#--------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                rules, and regulations of
#                IBM Global Services
#                Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                described in the IBM Customer Agreement and
#                relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                transfer any rights for replication or distribution
#                of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                course of system maintenance.
#                       2. Copying the program to a similar machine
#                within the same enterprise.
#
#               The customer agrees to restrict access to this
#                program as they would their own proprietary code,
#                and to notify IBM should unauthorized distribution
#                occur.
#
```

FIG. 12A

```
#              This program is distributed on an "as is" basis,
#                no warranty is expressed or implied.
#
#-------------------------------------

#-------------------------------------
#
# Description: Provides capability to perform split mirror backups.
#                  Part of FSCPBK package.
# Operational Environment: AIX V4 and ADSM V3.1
# Input:
# Output:
# Return Value:
# Comments:
#
#
#-------------------------------------

#-------------------------------------
#
# Version History: None
#
#-------------------------------------

#-------------------------------------
#
# Environmental Variables
#
#-------------------------------------
# Constants
bar='=================================================
'
wire='-----------------------------------------'

# Variables
numeric_date=$(date +%m%d%y)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i merge_return_code
typeset -i retain_days=90
typeset -i in_retain_days
typeset -i copies
typeset -i ncrement
typeset -i mount_fs_test
invoked_name=$0
script_name=${invoked_name##*/}
user_id=$(whoami)
desc='ADSM Archive at '$text_date
level=0
use_tape=0
```

FIG. 12B

```
# Process Control Variables
l_flag=0
L_flag=0
r_flag=0
d_flag=0

# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
default_backup_device=/dev/rmt0.1
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab
audit_file=/etc/fscpbktab.audit
lock_file=/var/locks/fscpbktab

#---------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return. It completely exits.
#
#---------------------------------
show_usage ()
{
 print -u2 "     "
 print -u2 "Usage: fscpbk_ack.ksh [-d device] [-l directory] [-r days]"
 print -u2 "     "
 print -u2 "     -d device    Backup output device."
 print -u2 "                  Default is" $default_backup_device
 print -u2 "     "
 print -u2 "     -l directory    Log output directory."
 print -u2 "                  Default is" $default_log_dir
 print -u2 "     "
 print -u2 "     -r days    Log retention period."
 print -u2 "                  Default is" $retain_days
 print -u2 "     "
 exit 2
}
```

FIG. 12C

```
#-------------------------------------
#
# Korn Shell Settings
#
#-------------------------------------
#set -o errexit    # Turn on error trapping and error exit mode
#set -o noclobber    # Prevent overwriting of existing files
#set -o noexec    # Perform syntax checking without execution
#set -o nolog    # Prevents storing function defs in history file
#set -o xtrace    # Turn on debug mode

#-------------------------------------
#
# Main Routine
#
#-------------------------------------
#
# Test for any passed parameters.
#if [ $? != 0 ]
#then
#    show_usage
#fi
#
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts d:l:r# c
do
  case $c in
  d)    # Set up the -d flag
        d_flag=1
        in_backup_device=$OPTARG;;
  1)    # Set up the -l flag
        l_flag=1
        log_dir=$OPTARG;;
   r)   # Set up the -r flag
        r_flag=1
        in_retain_days=$OPTARG;;
    :)  show_usage;;
   \?) show_usage;;
   esac
done
shift $((OPTIND-1))

# Deal with invocation errors
if [[ $user_id ! = root ]] then
      show_usage
fi

# Locate target file or device for backup images
if [[ $d_flag -eq 1 ]]; then
```

FIG. 12D

```
    if [[ $in_backup_device = /dev/rmt[0-9]* ]]; then  # Test if target is tape drive
       use_tape=1
       if [[ -c tin backup device ]]; then # Test if tape drive exists
           device=$in_backup_device
       else
           print -u2 "\nNonexistent tape drive" $in_backup_device
           show-Usage
       fi
    else    # Should we check to make sure some disk device not chosen?
           device=$ in_backup_device
    fi
else
   device=$default_backup_device
fi


# Configure Logging
if [[ $l - flag -eq 1 ]]; then
       log_file=$in_log_dir/$default_log_file
       mkdir -p $in_log_dir 2>/dev/null    #Create new log directory
else
       log_file=$default_log_dir/$default_log_file
       mkdir -p $default_log_dir 2>/dev/null  # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
       retain_days=$in_retain_days
fi

# Clear old logs
 find $log_dir -name "$script_name*" -mtime $retain_days -exec rm {}\;

# Create new log file
exec 3>> $log_file # Open log file for writing

print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log            ="
print -u3 "=                                               ="
print -u3 "= Created by script:" $script_name
print -u3 "=       on system:" $(hostname)
print -u3 "=       at        :" $(date)
print -u3 "=                                               ="
print -u3 "================================================="
```

FIG. 12E

```
# Perform Work

#  Test for existing table file
if [[ ! (-r $config_file) ]]; then
     print -u2 "Fatal Table error. Table file" $config_file "not found."
     print -u3 "Fatal Table error. Table file" $config_file "not found."
     exec 3<&-
     exit 99
fi

# Test for existing table audit file
if [[! (-r $audit_file) ]]; then
     print -u2 "Fatal Table error. Table file check program must be run."
     print -u3 "Fatal Table error. Table audit file" $audit_file "not found."
     exec 3<&-
     exit 97
fi

# Test for table file audit indicating syntax check since last edit


current_Y=$(date +%Y)

audit_stamp=$( head -1 $audit_file | awk '{ print $1 }')

#  Check for colon and thus time instead of year on file datestamp
ntest=$(ls -l $config_file | awk '{ print $8 }' | grep : | wc -l)
if [[  $ntest -eq 1 ]]; then
     edit_year=$current_Y
else

     edit_year=$(ls -l $config_file | awk '{ print $8 }')
fi


edit_month_text=$(ls -l $config_file | awk '{ print $6 }')
edit_day=$(ls -l $config_file | awk '{ print $7 }')
edit_hour=$(ls -l $config_file | awk '{ print $8 }' | cut -f 1 -d :)
edit_minute=$(ls -l $config_file | awk '{ print $8 }' | cut -f 2 -d :)

# Determine month number from month name
case $edit_month_text in
Jan) edit_month=01;;
Feb) edit_month=02;;
Mar) edit_month=03;;
Apr) edit_month=04;;
May) edit_month=05;;
Jun) edit_month=06;;
Jul) edit_month=07;;
```

FIG. 12F

```
Aug)   edit_month=08;;
Sep)   edit_month=09;;
Oct)   edit_month=10;;
Nov)   edit_month=11;;
Dec)   edit_month=12;;
*)   print -u2 "Fatal Table error. Table file date read error."
     print -u3 "Fatal Table error. Table file date read error."
     exec 3<&-
     exit 98;;
esac


edit_stamp=$edit_year$edit_month$edit_day$edit_hour$edit_minute

# Test for table file audited since last editing
if [[ $audit_stamp -le $edit_stamp ]]; then
    print -u2 "Fatal Table error. Table file edited since last checked."
    print -u3 "Fatal Table error. Table file edited since last checked."
    exec 3<&-
    exit 97
fi




# Table file format
# Format: bc:pfs:plv:c:afs:alv                =
# xb:/home:hd1:2:/alt/home:/altlvh            =


ncrement=0
return_code=0
# Cycle through filesystems and mount unmounted ones
for fs_line in $(cat $config_file | grep -v ^#)
do
   action=$(print $fs_line | cut -f 1 -d :)
   fs_prime=$(print $fs_line cut -f 2 -d :)
   lv_prime=$(print $fs_line cut -f 3 -d :)
   copies=$(print $fs_line | cut -f 4 -d :)
   target_fs=$fs_prime
   if [[ $action != no ]]; then
       if [[ $copies -gt 1 ]]; then
           target_fs= $(print $fs_line I cut -f 5 -d :)
       fi
```

FIG. 12G

```
# Check to see if target filesystem is mounted
      mount_fs_test=$(mount | grep "$target_fs | wc -l)
# If not mounted, mount as readonly for backups
      if [[ $mount_fs_test -ne 1 ]]; then
           mount -o ro $target_fs >>$log_file 2>>$log_file
           return_code=$?
# Test for unsuccessful readonly filesystem mount
           if [[ $return_code -ne 0 ]]; then
# If still unsuccessful, then perform filesystem check (presume dirty superblock)
                print -u3 "Performing fsck on filesystem" $target_fs
                fsck -p $target_fs >>$log_file 2>>$logfile
                mount -o ro $target_fs 2>>$log_file
           fi
      fi
  fi

done
return_code=0
merge_return_code=0

#  Put Table File at start of tape to serve as tape TOC
if [[ $use_tape -eq 1 ]]; then
     cp /etc/fscpbktab .
     echo "./fscpbktab"| backup -ipqf $device
     rm ./fscpbktab
fi

# Cycle through filesystems and perform backups and merges
for fs_line in $(cat $config_file | grep -v ^#)
do
     action=$(print $fs_line | cut -f 1 -d :)
     fs_prime=$(print $fs_line | cut -f 2 -d :)
     lv_prime=$(print $fs_line | cut -f 3 -d :)
     copies=$(print $fs_line | cut -f 4 -d :)
     target_fs=$fs_prime
     print -u3 $action $fs_prime $lv_prime $copies
     if [[ $action != no ]]; then
#         Select to backup alternate mirror fs if mirroring on
          if [[ $copies -gt 1 ]]; then
               fs_alt=$(print $fs_line cut -f 5 -d :)
               lv_alt=$(print $fs_line cut -f 6 -d :)
               target_fs=$fs_alt
               print -u3 $action $fs_prime $lv_prime $copies $fs_alt $lv_alt
          fi
```

FIG. 12H

```
      mount_fs_test=$(mount | grep "$target_fs" | wc -l)
#      Test for filesystem STILL not mounted
      if [[ $mount_fs_test -eq 1 ]]; then
   case $action in
   no) # Perform no backup action
      print -u3 "No backup performed on filesystem" $target_fs;;
   xb) # Perform AIX Level 0 filesystem backup
      print -u3 "Starting AIX Level 0 backup on filesystem" $target_fs "at" $(date)
      backup -$level -u -f $device $target_fs
      return_code=$return_code+$?
      print -u3 "Completed AIX Level 0 backup on filesystem" $target_fs "at" $(date);;
   as) # Perform ADSM Selective filesystem backup
      print -u3 "Starting ADSM Selective backup on filesystem" $target_fs "at" $(date)
      dsmc sel "$target_fs/*" >$work_file1
      return_code=$return_code+$?
      cat $work_file1 >>$log_file
      print -u3 "\n ------------------------------"
      print -u3 "Completed ADSM Selective backup on filesystem" $target_fs "at" $(date);;
   ai) # Perform ADSM Incremental filesystem backup
      print -u3 "Starting ADSM Incremental backup on filesystem" $target_fs "at" $(date)
      dsmc i $target_fs >$work_file1
      return_code=$return code+$?
      cat $work_file1 >>$log_file
      print -u3 "\n------------------------------"
      print -u3 "Completed ADSM Incremental backup on filesystem" $target_fs_prime "at"
$(date);;
   aa) # Perform ADSM Archive filesystem archive
      print -u3 "Starting ADSM Archive on filesystem" $target_fs "at" $(date)
      dsmc archive $target_fs/ -des="$desc" >$work_file1
      return_code=$return_code+$?
      cat $work_file1 >>$log_file
      print -u3 "\n ------------------------------"

      print -u3 "Completed ADSM Archive on filesystem" $target_fs "at" $(date);;
  esac
#    Merge split filesystems if mirrored
#     NOTE!!: This section is duplicated in the fscpbk_merge.ksh
#         script. Any changes anywhere in this script should
#         probably be duplicated in that script!
#
      if [[ $copies -gt 1 ]]; then
         merge_fs_copy.ksh -p $fs_prime -s $fs_alt
#          merge_return_code=$merge_return_code+$?
#          fs_alt=$(print $fs_line | cut -f 5 -d :)
#          lv_alt=$(print $fs_line | cut -f 6 -d :)
#          target_fs=$fs_alt
      fi
```

FIG. 12I

```
#!/bin/ksh
#################################################################
#
# fscpb_merge.ksh
#               Version 0.01
#                       Runs various AIX commands to merge
#                         filesystems
#                       Assembled by Carl Gusler
#                        IBM Global Services
#                        IBM Austin
#                        cgusler@us.ibm.com
#
#                               (With help from many friends)
#
#                       Copyright IBM 1996, 1997, 1998, 1999
#                       Controlled Distribution
#                       Protected under the procedures, processes, rights
#                        rules, regulations, and retributions of
#                        IBM Global Services
#                        Intellectual Capital Management
#
#
#################################################################

#--------------------------------
#
# Copyright Information: Copyright IBM 1998
#               Controlled Distribution
#               Protected under the procedures, processes, rights
#                rules, and regulations of
#                IBM Global Services
#                Intellectual Property Management
#
#               This program is an IBM Type II Deliverable as
#                described in the IBM Customer Agreement and
#                relevant IBM services contracts.
#
#               IBM retains all rights to this program and does not
#                transfer any rights for replication or distribution
#                of this program except for the following:
#                       1. Backup/archive copies taken as a normal
#                course of system maintenance.
#                       2. Copying the program to a similar machine
#                within the same enterprise.
#
#              The customer agrees to restrict access to this
#                program as they would their own proprietary code,
#                and to notify IBM should unauthorized distribution
#                occur.
```

FIG. 13A

```
#
#               This program is distributed on an "as is" basis,
#                  no warranty is expressed or implied.
#
#-----------------------------------

#-----------------------------------
#
# Description: Remerges filesystems split from mirrored LVs.
#         A cleanup utility for problem times with FSCPBK scripts
# Operational Environment: AIX V4
# Input:
# Output:
# Return Value:
# Comments: NOTE!!: This script is an excerpt of the fscpbk_back.ksh
#                  script. If that script is edited, this one
#                  should probably be edited to match.
#
#-----------------------------------

#-----------------------------------
#
# Version History: None
#
#-----------------------------------

#-----------------------------------
#
# Environmental Variables
#
#-----------------------------------
# Constants
bar='=================================================
'
wire='-----------------------------------'

# Variables
numeric_date=$(date +%m%d%y)
text_date=$(date +%d%b%Y)
typeset -i return_code
typeset -i merge_return_code
typeset -i retain_days=90
typeset -i in_retain_days
typeset -i copies
typeset -i ncrement
typeset -i mount_fs_test
invoked_name=$0
script_name=${invoked_name##*/}
user_id=$(whoami)
desc='ADSM Archive at'$text_date
level=0
```

FIG. 13B

```
# Process Control Variables
l_flag=0
L_flag=0
r_flag=0
d_flag=0

# Files

default_log_dir=/var/adm/scriptlogs
default_log_file=$script_name.$text_date
default_backup_device=/dev/rmt0.1
work_file1=/tmp/$script_name.$text_date.work1
work_file2=/tmp/$script_name.$text_date.work2
config_file=/etc/fscpbktab
audit_file=/etc/fscpbktab.audit
lock_file=/var/locks/fscpbktab

#-----------------------------------
#
# Function: show_usage
#       Description: Displays command usage syntax and exits
#       Input: None
#       Output: Usage message to standard error
#       Return Value: 2
#       Note: This function does not return. It completely exits.
#
#-----------------------------------
show_usage ()
{
    print -u2 "       "
    print -u2 "Usage: fscpbk_merge.ksh [-l directory] [-r days]
    print -u2 "       "
    print -u2 "       -l directory    Log output directory."
    print -u2 "                       Default is" $default_log_dir
    print -u2 "       "
    print -u2 "       -r days         Log retention period."
    print -u2 "                       Default is" $retain_days
    print -u2 "       "
    exit 2
}

#-----------------------------------
#
# Korn Shell Settings
#
#-----------------------------------
#set -o errexit    # Turn on error trapping and error exit mode
#set -o noclobber    # Prevent overwriting of existing files
#set -o noexec    # Perform syntax checking without execution
```

FIG. 13C

```
#set -o nolog        # Prevents storing function defs in history file
#set -o xtrace       # Turn on debug mode

#-------------------------------------
#
# Main Routine
#
#-------------------------------------
#
# Test for any passed paramaters.
#if [ $? != 0 ]
#then
#   show_usage
#fi
#
log_dir=$default_log_dir
# Parse Command Line Arguments into Variables
while getopts l:r# c
do
    case $c in
    l)  # Set up the -l flag
        l_flag=1
        log_dir=$OPTARG;;
    r)  # Set up the -r flag
         r_flag=1
         in_retain_days=$OPTARG;;
     :)  show_usage;;
     \?) show_usage;;
     esac
done
shift $((OPTIND-1))

# Deal with invocation errors
if [[ $user_id != root ]]; then
     show_usage fi


# Configure Logging
if [[ $l_flag -eq 1 ]]; then
     log_file=$in_log_dir/$default_log_file
     mkdir -p $in_log_dir 2>/dev/null    #Create new log directory
else
     log_file=$default_log_dir/$default_log_file
     mkdir -p $default_log_dir 2>/dev/null # Create default log directory
fi

if [[ $r_flag -eq 1 ]]; then
     retain_days=$in_retain_days
fi
```

FIG. 13D

```
# Clear old logs
find $log_dir -name "$script_name*" -mtime $retain_days -exec rm {}\;

# Create new log file
exec 3>> $log_file # Open log file for writing

print -u3 "\n================================================="
print -u3 "=                                               ="
print -u3 "= Systems Management Transaction Log            ="
print -u3 "=                                               ="
print -u3 "=  Created by script:" $script_name
print -u3 "=        on system:" $(hostname)
print -u3 "=        at    :" $(date)
print -u3 "=                                               ="
print -u3 "================================================="

# Perform Work
#  Comments: NOTE!!: This script is an excerpt of the fscpbk_back.ksh
#                    script. If that script is edited, this one
#                    should probably be edited to match.
#

#  Test for existing table file
if [[  ! (-r $config_file) ]]; then
    print -u2 "Fatal Table error. Table file" $config_file "not found."
    print -u3 "Fatal Table error. Table file" $config_file "not found."
    exec 3<&-
    exit 99
fi

# Test for existing table audit file
if [[ ! (-r $audit_file) ]]; then
    print -u2 "Fatal Table error. Table file check program must be run."
    print -u3 "Fatal Table error. Table audit file" $audit_file "not found."
    exec 3<&-
    exit 97
fi

# Test for table file audit indicating syntax check since last edit

current_Y=$(date +%Y)

audit_stamp=$( head -1 $audit_file | awk'{ print $1 }')

#  Check for colon and thus time instead of year on file datestamp
ntest=$(ls -l $config_file | awk'{ print $8 }' | grep : | wc -l)
if [[ $ntest -eq 1 ]]; then
    edit_year=$current_Y
```

FIG. 13E

```
else

   edit_year=$(ls -l $config_file | awk '{ print $8 }')
fi


edit_month_text=$(ls -l $config_file | awk '{ print $6 }')
edit_day=$(ls -l $config_file | awk '{ print $7 }'
edit_hour=$(ls -l $config_file | awk '{ print $8 }' | cut -f 1 -d :)
edit_minute=$(ls -l $config_file | awk '{ print $8 }' | cut -f 2 -d :)

#  Determine month number from month name
case $edit_month_text in
Jan)   edit_month=01;;
Feb)   edit_month=02;;
Mar)   edit_month=03;;
Apr)   edit_month=04;;
May)   edit_month=05;;
Jun)   edit_month=06;;
Jul)   edit_month=07;;
Aug)   edit_month=08;;
Sep)   edit_month=09;;
Oct)   edit_month=10;;
Nov)   edit_month=11;;
Dec)   edit_month=12;;
*)     print -u2 "Fatal Table error. Table file date read error."
       print -u3 "Fatal Table error. Table file date read error."
       exec 3<&-
       exit 98;;
esac


edit_stamp=$edit_year$edit_month$edit_day$edit_hour$edit_minute

# Test for table file audited since last editing
if [[ $audit_stamp -le $edit_stamp ]]; then
    print -u2 "Fatal Table error. Table file edited since last checked."
    print -u3 "Fatal Table error. Table file edited since last checked."
    exec 3<&-
    exit 97
fi


# Table file format
# Format: bc:pfs:plv:c:afs:alv                    =
# xb:/home:hd1:2:/alt/home:/altlvh                  =
```

FIG. 13F

```
ncrement=0
return_code=0
merge_return_code=0

# Cycle through filesystems and perform merges
for fs_line in $(cat $config_file | grep -v ~#)
do
   action=$(print $fs_line | cut -f 1 -d :)
   fs_prime=$(print $fs_line | cut -f 2 -d :)
   lv_prime=$(print $fs_line | cut -f 3 -d :)
   fs_alt=$(print $fs_line | cut -f 5 -d :)
   lv_alt=$(print $fs_line | cut -f 6 -d :)
   copies=$(print $fs_line | cut -f 4 -d :)
   target_fs=$fs_prime
   print -u3 $action $fs_prime $lv_prime $copies
   if [[ $action != no ]]; then

#     Merge split filesystems if mirrored
      if [[ $copies -gt 1 ]]; then
         merge_fs_copy.ksh -p $fs_prime -s $fs_alt
         merge_return_code=$merge_return_code+$?
      fi
   fi
done



exec 3<&-

# Test for unsuccessful filesystem merges
if [[ $merge_return_code -ne 0 ]]; then
    exit 20
fi

# Remove lock on table file
rm $lock_file 2>/dev/null
chmod 644 $config_file

exit 0
-
```

FIG. 13G